VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2020 Tomáš Bičák

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

JEDNOÚČELOVÝ VOIP KOMUNIKÁTOR
DEDICATED VOIP COMMUNICATOR

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Tomáš Bičák**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Tomáš Caha**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Telekomunikační a informační systémy**

Ústav telekomunikací

Student: Tomáš Bičák ***ID:*** 155737
Ročník: 3 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Jednoúčelový VoIP komunikátor

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s technologií VoIP a protokolem SIP. Vytvořte prototyp zařízení, které po stisknutí tlačítka naváže hlasové spojení s určitým účastníkem. Pro VoIP komunikaci využijte vybraného poskytovatele (např. sip2sip.info). Vytvořený kód vystavte pod licencí MIT na GitHub.

DOPORUČENÁ LITERATURA:

[1] Linux Dokumentační projekt. 4. vyd. Computer Press, 2008. 1336 s. ISBN: 978-80-251-1525-1.

[2] PILGRIM, M. Ponořme se do Python(u) 3. CZ.NIC, 2010. 435 s. ISBN: 978-80-904248-2-1.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 8.6.2020

Vedoucí práce: Ing. Tomáš Caha

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

ABSTRAKT

Bakalářská práce se věnuje návrhu jednoduchého VoIP komunikátoru a to na Raspberry Pi. K Raspberry Pi je připojena externí zvuková karta ReSpeaker 2 Mics Pi HAT. Obslužný program je napsán v programovacím jazyce Python.

KLÍČOVÁ SLOVA

SIP, Raspberry Pi Zero WH, ReSpeaker 2 Mics Pi HAT, VoIP, PJSIP

ABSTRACT

The bachelor's thesis deals with the design of a simple VoIP communicator on a Raspberry Pi. An external ReSpeaker 2 Mics Pi HAT sound card is connected to the Raspberry Pi. The utility is written in the Python programming language.

KEYWORDS

SIP, Raspberry Pi Zero WH, ReSpeaker 2 Mics Pi HAT, VoIP, PJSIP

BIČÁK, Tomáš. *Jednoduchý VoIP komunikátor*. Brno, 2020, 61 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Tomáš Caha

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Jednoduchý VoIP komunikátor“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Tomáši Cahovi, za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

Tato bakalářská práce navazuje na semestrální projekt, kde se hledaly a zkoušely programy na VoIP (Voice over Internet Protocol) technologie na zařízení Raspberry Pi Zero WH.

Technologie VoIP je technologie, která umožňuje přenos digitalizovaného hlasu prostřednictvím počítačové sítě. Technologie VoIP využívá sadu protokolů a kodeků ke komunikaci. K navázání a ukončení hovorů ve VoIP se používá protokol SIP (Internet Protocol). Samotný protokol SIP je spojen s dalšími důležitými protokoly a to jsou protokoly RTP (Real-time Transport Protocol), RTCP (RTP Control Protocol) a SDP (Session Description Protocol). Mezi nejčastěji používané kodeky ve VoIP patří G.711, G.722 a atd.

K samotnému testování VoIP komunikátoru je použito zařízení Raspberry Pi Zero, které vyvinula britská společnost Raspberry Pi Foundation v roce 2017. Raspberry Pi je nejrozšířenější jednodeskový počítač na světě. Je využívan jak pro studijní účely tak i v domácnostech na jednoduchou automatizaci nebo jako levný počítačový server na kterém jsou spuštěny potřebné služby.

Na RaspberryPi je použit programovací jazyk Python. Programovací jazyk Python je vysokoúrovňový skriptovací programovací jazyk, který byl vyvinut v roce 1991. Nabízí dynamickou kontrolu datových typů a podporuje různé programovací styly včetně objektově orientovaného, imperativního, procedurálního nebo funkcionálního. Python byl vyvíjen jako open source projekt, který nabízí zdarma instalační balíky pro většinu běžných operačních systémů.

Tato bakalářská práce se věnuje sestavením jednoduchého VoIP komunikátoru, na jednodeskové počítači je nainstalován program na VoIP komunikaci. Tento program je ovládán pomocí skriptů napsaných v programovacím jazyce Python. Raspberry Pi je osazeno speciálním zařízením, které slouží jako zvuková karta a obsahuje tlačítko. Pomocí tlačítka se ovládá celé zařízení.

Dokumentace je rozdělena do kapitol. První kapitola se zabývá teoretickým popisem zařízení Raspberry Pi a všech jeho součástí. Dále je zde popsána technologie VoIP a WIFI, protokol SIP, RTP, RTCP, SDP, standart H.323 a QoS. Druhá kapitola se věnuje postupu instalace Raspberry Pi, zprovoznění zvukové karty ReSpeakr 2 Mics Pi HAT, stažením a kompilací PJSIP. V poslední části této kapitoly je návrh obslužného programu. Poslední kapitola se věnuje zhodnocením celé bakalářské práce.

1 Použitý hardware a software

Tato část se zabývá teoretickým popisem všech částí zařízení na kterém je bakalářská práce realizována. Následně je zde popsána technologie VoIP a protokoly, které jsou s ní spojeny. To jsou protokoly SIP, RTP, RTCP a SDP. Následně je zde popsán nejčastěji používaný kodek G.711.

1.1 Vývoj zařízení Raspberry Pi

Raspberry Pi bylo vyvinuto britskou společností Raspberry Pi Foundation v roce 2012 s cílem podpořit výuku informatiky na školách a seznámit studenty s tím, jak můžou počítače ovládat různá zařízení. Raspberry Pi je název malého jednodeskového počítače. Existují i jiné jednodeskové počítače, které vychází z Raspberry Pi, jsou to například Banana Pi, Khadas, Odroid a další. Dále je v textu Raspberry Pi nahrazeno jeho zkratkou RPi.

Někdy se RPi přirovnává k Arduinu, což je také jednodeskový počítač, který je založen na procesorech ATmega od společnosti Atmel.

Na trhu existuje více variant RPi, které se od sebe liší velikostí a použitým procesorem. Na RPi je použit mikroprocesor z rodiny ARM(Ashton Raggatt McDougall). Výkon RPi v dnešní době dosahuje výkonu srovnatelného s dnešními mobilními telefony. Lze říci, že nejmodernější RPi 4 dosahuje výkonu přenosného počítače. V tabulce 1.1 na straně 17 je porovnání všech vlastností minulých a současných verzí RPi. [3]

1.1.1 Využití Raspberry Pi

Nejčastěji se Raspberry Pi používá jako základ pro automatizaci například domácností, nebo se používá v počítačových sítích jako uložiště dat nebo jako server, na kterém běží různé služby (DHCP (Dynamic Host Configuration Protokol),DNS (Domain Name System),Pi-Hole (Blokování internetové inzerce) a atd.). RPi se dá v domácnosti využít jako multimediální centrum s operačním systémem LibreELEC. Jako automatizační jednota se může využít na dálkové ovládání žaluzií, garážových vrat. Nebo může být připojen k 3D tiskárně a sloužit jako printserver. Lze RPi využít jako Web server, který pracuje v lokální síti nebo je připojen k internetu. Na internetu je mnoho projektů realizovaných pomocí RPi.

1.1.2 Raspberry Pi Zero

RPi Zero je nejmenší, nejlevnější a nejúspornější z RPi. Toto RPi se hodí k využití, tam, kde je potřeba málo místa s velkým potenciálem. Zařízení je poháněno

mikroprocesorem o taktu 1GHz. Na trhu existují tři varianty RPi Zero a to v první variantě bez GPIO sběrnice a WiFi modulu. Druhá varianta obsahuje WiFi modul a neobsahuje také sběrnici GPIO. A poslední varianta obsahuje GPIO sběrnici a WiFi modul. [4]

1.1.3 Raspberry Pi 3

RPi model 3 se na trhu objevuje ve varientě B a B+, která se od sebe liší v použité síťové kartě a ve výkonu procesoru. První na trh se dostala varianta RPi 3 B, která měla jako první 64-bitový mikroprocesor s taktem 1,2 GHz a obsahovala WiFi kartu se standartem 802.11n. Novější model RPi 3 B+, tak má vyšší výkon procesoru (1,4 GHz), síťová karta využívá maximální rychlosti přenosu 1 Gb/s a WiFi karta podporuje standart 802.11ac. Oproti původní variantě je možné už připojit Raspberry Pi PoE HAT, který dokáže napájet RPi pomocí ethernetového portu. [5]

1.1.4 Raspberry Pi 4

RPi 4 je nejnovější varianta Raspberry, která přišla na trh s novými použitými konektory. Změnil se konektor napájení, původně RPi využívalo microUSB konektor a nově je využíváno USB C. Nová verze má dva video výstupy. Změnil se video výstup, který byl konektor HDMI (High-Definition Multimedia Interface) na microHDMI. Původní verze měla 4 konektory USB ve standartu 2.0 u nové verze byly 2 konektory USB 2.0 nahrazeny standartem 3.0. Nově je nabízeno RPi ve variantách s různou velikostí operační paměti, a to o velikostech 2, 4 a 8 Gb. RPi 4 má oproti variantě 3 B+ větší takt mikroprocesoru (1,5 GHz) a podporuje maximální rozlišení video výstupu 4Kp60. [6]

1.1.5 Sběrnice

RPi nedisponuje zabudovaným A/D převodníkem. Proto je nutné použít rozšiřující karty, které disponují komunikačními sběrnicemi I2C (Inter Integrated Circuit) a nebo SPI (Serial Peripheral Interface). Tyto sběrnice komunikůjí s RPi pomocí GPIO konektoru.

Sběrnice I2C

I2C je sběrnice vyvinutá firmou Philips, která se používá na připojení periférií. Sběrnice I2C obsahuje jednu řídící jednotku (master) a jednotky slave. Sběrnice využívá tři vodiče, z toho dva jsou datové vodiče a třetí vodič je sdílená zem. Datové vodiče jsou označovány SDA (Serial Data Line) a SCL (Serial Clock Line). Datový vodič

SDA je vodič který slouží k odesílání dat. Vodič SCL slouží k odesílání synchronizačního hodinové signálu. Tento signál vysílá výhradně master jednotka, ostatní jednotky slave se synchronizují se s tímto signálem. Datové vodiče bývají připojeny pomocí puul-up rezistorů k napájecímu napětí. Každá jednotka slave má unikátní 10-ti bitovou adresu. [13]

Sběrnice SPI

Sběrnice SPI je jednodušší pro implementaci než sběrnice I2C. Stejně jako u sběrnice I2C je jedna master jednotka a více jednotek slave. Propojení mezi těmito jednotkami je pomocí vodičů SCK (Serial Clock), MOSI (Master Out, Slave In), MISO (Master In, Slave Out) a SS (Slave Select). Vodič SCK slouží k přenášení hodinového signálu. Další vodič MOSI slouží k přenášení dat směrem od master zařízení k slave jednotce. MISO vodič slouží k přenášení dat v opačném směru než MOSI. Poslední vodič SS slouží k výběru slave jednotky. Pro každé zařízení slave může být využit zvlástní vodič SS, nebo tyto jednotky slave mohou být napojeny na jeden vodič SS. [14]

1.1.6 Rozšiřující moduly

Raspberry Pi je ve světě dost rozšířené a existují na něj různé rozšiřující moduly, které vytvořila společnost Raspberry Pi Foundation. Na trhu existují moduly, které vyvíjí různé společnosti. Na RPi je možné dokoupit různé moduly, které rozšíří daná zařízení o další možnosti využití. Lze dokoupit různé dotykové obrazovky různých velikostí jak od společnosti Raspberry, tak i od ostatních výrobců. Dále se dá dokoupit zvuková karta,tepelný snímač a automatizační karta, která obsahuje relé. Z RPi se dá vytvořit pomocí těchto rozšiřujících modulů i 3D tiskárna, nebo domácí multimediální centrum s televizní kartou.

1.1.7 Výběr Raspberry Pi a rozšiřujícího modulu

Pro účely bakalářské práce byl vybrán nejmenší jednodeskový počítač z rodiny RPi, který je vyobrazen na obrázku 1.1 na straně 18. Z toho důvodu, že je energeticky nenáročný. Jelikož RPi neobsahuje zvukový výstup, tak byla vybrána externí zvuková karta ReSpeaker 2 Mics Pi HAT od společnosti ReSpeaker.

1.1.8 Operační systém Raspberry Pi

Operační systém pro RPi je speciálně upravený operační systém pro architekturu ARM. Z toho důvodu nelze na RPi nainstalovat jiný operační systém. Operační systémy, které podporují architekturu ARM jsou optimalizovány, tak aby bylo dosaženo

co nejvyššího výkonu a stability sytému. Systémy, které jsou schopny pracovat na RPi jsou na bázi Linuxu, ale lze i použít operační systém od společnosti Microsoft. [3]

Raspbian

Raspberry Pi OS neboli Raspbian je operační systém založen na linuxové distribuci Debian. Systém byl vydán v roce 2012 s velkým množstvím doporučených balíčků, které měli za úkol usnadnit a zpříjemnit práci uživatele. Tento operační systém je nejvíce rozšířený mezi uživateli RPi pro svoji jednoduchost a přehlednost. Na oficiálních stránkách je více verzí tohoto operačního systému. Lze ho stáhnout ve verzi Lite, která neobsahuje grafické rozhraní a celé ovládání se provádí pomocí příkazového řádku. Tato verze je doporučována pro zkušenější uživatele. Dále je dostupná verze, která je s grafickým rozhraním a poslední verze je s doporučeným softwarem. [3]

Ubuntu Mate

Ubuntu Mate je operační systém založen na uživatelsky nejrozšířenějším operačním systému Ubuntu. Tato verze operačního systému pro RPi je speciálně upravená, tak aby poskytovala uživateli kompletní a známé prostředí pro stolní počítače. Tento operační systém bohužel nejde spustit na zařízeních RPi ZERO a RPi 1 model A i B. [7]

Kali Linux

Kali linux je linuxová distribuce odvozena od operačního systému Debian. Je navržen pro digitální forenzní analýzu a penetrační testy na prolomení hesel. Má více než 600 předinstalovaných programů na testování.. Tento operační systém byl primárně vyvinut na testování bezpečnosti počítačových sítí. Doporučuje se pro velmi zkušené uživatele.

Arch Linux

Arch Linux je primárně vyvíjen pro procesory s architekturou x86_64 a i386. Tento operační systém klade důraz na minimalistický a snadno přizpůsobitelný systém. Lze ho přizpůsobit k jakémkoli použití. Doporučuje se především pro zkušené uživatele. [12]

OSMC

OSMC (Open Source Media Center) je bezplatný multimediální přehrávač založený na operačním systému Linux. Byl představen již v roce 2004 a umožňuje přehrávat videa z lokálního úložiště dat nebo z internetu. Tento operační systém je primárně určen pro zařízení Vero, ale dá se využít i na RPi. Je založen na multimediálním přehrávači KODI. [8]

LibreELEC

LibreELEC je operační systém pro RPi, který z tohoto zařízení vytvoří multimediální centrum. Je postaven na multimediálním přehrávači KODI. Celý systém je dobře optimalizovaný a je možné ho ovládat pomocí dálkového ovladače od televize, pokud je RPi spojeno pomocí HDMI kabelu s televizí. Na tento operační systém slouží rozšiřující balíčky. [9]

PiNet

PiNet je operační systém, který je navržen pro školy a školící organizace k nastavení a správě sítí Raspberry Pi. Slouží k replikaci systémů, které existují pro sitě Microsoft. Všechen software je volně dostupný a je vytvořen pedagogy z celého světa. [10]

RISC OS

Operační systém RISC OS není postaven na jádru operačního systému Linux nebo Unix. Považuje se za první operační systém pro architekturu ARM již v roce 1987. Tento operační systém s aplikacemi dosahoval velikosti 6 MB. Původně byl vyvinut pro mnohem pomalejší procesory a se spojením s RPi dosahuje tento systém rychlé odezvy na požadavky uživatele.

Windows 10 IOT

Windows 10 IOT je operační systém od společnosti Microsoft. Tento operační systém je určen pro jednoúčelové využití. Lze ho využít v bankomatech, herních automatech, tiskových zařízeních a ve zdravotních přístrojích, které sdílí informace pomocí cloudových služeb. Z operačního systému Windows 10 používá jen jádro systému. Je bez grafického rozhraní a tento operační systém není prodáván koncovým spotřebitelům, ale je prodáván primárně OEM (Original Equipment Manufacture) výrobcům, kteří jej následně upravují a prodávají koncovým uživatelům.[11]

Tab. 1.1: Přehled všech verzí Raspberry Pi

Typ	Generace	Architektura	Procesor	SDRAM	USB	Síť	Výkon
Model A	1	ARMv6 (32-bit)	700 MHz single-core	256 MiB	1		300 mA
Model A	1+	ARMv6 (32-bit)	700 MHz single-core	512 MiB	1		200 mA
Model B	1	ARMv6 (32-bit)	700 MHz single-core	512 MiB	2	10/100 Mbit	700 mA
Model B	1+	ARMv6 (32-bit)	700 MHz single-core	512 MiB	4	10/100 Mbit	600 mA
Model B	2	ARMv7 (32-bit)	900 MHz 32-bit quad-core	1 GiB	4	10/100 Mbit	800 mA
Model B	3	ARMv8 (64/32-bit)	1,2 GHz 64-bit quad-core	1 GiB	4	10/100 Mbit, 802.11n	200 mA
Zero	PCB 1.3	ARMv6 (32-bit)	700 GHz single-core	512 MiB	1		160 mA
Zero	W	ARMv6 (32-bit)	1 GHz single-core	512 MiB	1	802.11n	160 mA
Model B	3+	ARMv8 (64/32-bit)	1,4 GHz 64-bit quad-core	1 GiB	4	10/100/1000 Mbit, 802.11ac	459 mA
Model B	4	ARMv8 (64-bit)	1,5 GHz 64-bit quad-core	1, 2, 4, 8 GiB	4	10/100/1000 Mbit, 802.11ac	3A

Obr. 1.1: Raspberry ZERO WH

1.2 SD karta

SD karta je paměťové zařízení, které slouží k ukládání dat. Existují různé standardy, které určují typy paměťových karet podle velikosti úložného místa, přenosové rychlosti zápisu a čtení. Paměťové karty typu SD jsou nejrozšířenějším typem na trhu, ale existují i jiné typy paměťových karet. Například jsou to paměťové karty typu Compact Flash (SF), Memory Stick, MultiMediaCard (MMC) nebo Smart Media. SD karty se rozdělují podle rozměru na SD, miniSD, microSD. Rychlosti se dělí na „Normal speed“, „High speed“ a „UHS-I“. Kapacita paměťových karet začíná na 16 MB po 2 TB, ale v dnešní době nejsou dostupné velikosti do 16 Mb do 2 GB. [16]

1.3 Universal Serial Bus (USB)

USB je univerzální sériová sběrnice, která je dostatečně všestranná pro jakékoliv využití. Tato sběrnice je velice rozšířená a nahrazuje ostatní typy sběrnic. Mezi základní zařízení, které využívají sběrnici USB je klávesnice, myš, USB sluchátka a mikrofony, tiskárny, mobily, kamery. [15]

1.3.1 Historie

USB vzniklo spolupráci různých firem. Nahrazuje používaný sériový port RS-232. USB sběrnice ulehčuje práci uživateli a má vyšší šířku sběrnice než sériový port RS-232. První specifikace byla stanovena v roce 1995, ale začala se používat až v roce 1998.

1.3.2 Zapojení

Standardní verze USB 2.0 má celkem 4 vodiče z toho je jeden napájecí, jeden GND a dva vodiče datové. Datové vodiče mají standartně barvu zelenou a bílou. Napájecí kabel má barvu červenou a GND má barvu černou. Kabely bývají stíněny hliníkovou fólií.

1.3.3 Generace USB

Tato sekce se zabývá rozdělením sběrnice USB podle standardů, jak byly uvedeny na trh.

USB 1.1

Ve verzi USB 1.1 existují pomalá (Low-Speed) zařízení s přenosovou rychlostí 1,5 Mbit/s (187,5 kB/s) a rychlá zařízení (Full-Speed) s rychlostí 12 Mbit/s (1,5 MB/s). USB 1.1 však nebylo schopno konkurovat vysokorychlostním rozhraním.

USB 2.0

V roce 1999 se začalo uvažovat o druhé generaci USB, která by byla použitelná i pro náročnější zařízení (např. digitální kamery). Tato nová verze, označovaná jako USB 2.0, přišla v roce 2000 a nabídla maximální rychlost 480 Mbit/s (60 MB/s) v režimu Hi-Speed, avšak zachovala zpětnou kompatibilitu s USB 1.1 (režimy Low-Speed a Full-Speed).

USB 3.1 generace 1

Třetí generace USB je někdy označována jako Superspeed USB a začala se rozšiřovat až v roce 2010. USB 3.1 disponuje více něž desetinásobnou přenosovou rychlostí oproti původnímu standartu. Přenosová rychlost dosahuje maximálně Gbit/s. Oproti USB 2.0 už nevyužívá 4 vodiče ale 9 vodičů. Počet datových vodičů se zvětšil ze 2 na 4. Zpětná kompatibilita se standardem USB 2.0 je zajištěna.

USB 3.1 generace 2

Generace 2 oproti první generaci nabízí zvětšenou maximální přenosovou rychlost a to 10 Gb/s. Díky této maximální rychlosti se vyrovná konkurenčnímu Thunderboltu od společnosti Apple. Zpětná kompatibilita se staršími generacemi je zajištěna.

USB-C

USB-C se rozměry blížit variantě microUSB, svými vlastnostmi konektoru Lightning od Applu. Stejně jako ten, půjde typ C USB 3.1 zasunout oběma směry navíc i vzhůru nohama. První zařízení s novým USB-C se začala objevovat v polovině roku 2014. Nový konektor nebude disponovat pouze datovou prostupností 10 Gb/s, ale může být použit i k napájení. Maximem je 5 A při 20 V, tedy 100 W (v porovnání se starší verzí je to až 40x více), což bude dostačovat i pro napájení většiny notebooků. Tento konektor je kompatibilní s: Linux, Windows 10, Windows 8.1, OS X, Android 6 a Chrome OS. USB typu C má na obou stranách stejný konektor skládající se z 24 kontaktů (12 shora, 12 zespoda). Díky již zmíněným zajímavostem bude USB-C používáno téměř ve všech zařízeních.

1.4 Rozhraní HDMI

HDMI může propojovat například satelitní přijímač, DVD (Digital Versatile Disc) přehrávač nebo videopřehrávač s kompatibilním zobrazovacím zařízením, jako například televizor s plazmovou obrazovkou pro systém domácího kina. HDMI podporuje přenos videa ve standardní, rozšířené nebo high-definition kvalitě, a až 8kanálový digitální zvuk. [17]

1.4.1 Historie

Vývoj HDMI 1.0 začal 16. dubna 2002 s cílem vytvořit zařízení zpětně kompatibilní s DVI, které se v tu dobu vyskytovalo na většině HD (High-Definition) televizorů a přehrávačů DVD. HDMI bylo vytvořeno k vylepšení DVI pomocí menšího konektoru s přidanou podporou pro přenos zvuku. Vzhledem k úspěšnosti na trhu se stal z HDMI celosvětový standard.

1.4.2 Specifikace

V současnosti existují čtyři typy konektorů HDMI: A, B, C a D. Typy A a B jsou definovány standardem 1.0, typ C je definován až od verze 1.3 a konektor typu D je definován od verze 1.4.

HDMI nedefinuje maximální délku kabelu. Jediným omezením je útlum signálu na kabelu. Délka kabelu závisí na konstrukci a kvalitě použitých materiálů. Od roku 2012 se kabely neoznačují číslem verze, ale slovně. [17]

1.4.3 Verze HDMI

V současné době existuje několik verzí HDMI. HDMI 1.0 byla představena v roce 2002 a umožňovala maximální propustnost 4.9 Gbit/s. Následná verze 1.1 byla představena v roce 2004 a přinášela podporu pro DVD-audio.

Potřeba nového zobrazovacího rozhraní vedl k vývoji verze HDMI 1.2, která byla představena v roce 2005. V této verzi se objevil poprvé HDMI konektor typu A pro PC. Dále je zde možnost převodu RGB (Reg-Green-Blue) do YCbCr. Později byla přidána podpora pro CEC (Consumer Electronics Control)

Verze HDMI 1.3 byla představena v roce 2006 se zvětšenou šířkou pásma. Tato verze přinesla možnost zvolit barevnou hloubku 30-bit, 36-bit a 48-bit. Dále se tu objevila možnost automatické zvukové synchronizace.

Verze HDMI 1.4 byla uvedena v roce 2009 a byla u ní zvýšena maximální propustnost na 10,2 Gb/s. Objevila se zde podpora pro 3D (Three-dimensional space) a byl přidán kanál pro Ethernet. Maximální rozlišení přenášeného videa se zvětšilo na 4096x2160 24Hz.

Verze HDMI 2.0 označována jako Premium HDMI high speed byla představena v roce 2013 u této verze byla přidána podpora formátu 21:6, maximální rozlišení se zvýšilo na 4K 60Hz, umožňuje podporo až pro 4 audio stopy.

Nejaktuálnější verze HDMI 2.1 označovaná jako HDMI ultra high speed byla představena v roce 2017. U této verze se zvýšila maximální propustnost na 48 Gb/s. Podporuje vyšší video rozlišení a obnovovací rychlosti. Podporuje také dynamické formáty HDR (High Dynamic Range Imaging). Tento standart HDMI 2.1 umožňuje podporu 8K 60Hz a 4K 120Hz a maximální rozlišení 10K.

1.4.4 Výhody HDMI

- Přenos nekomprimovaných dat.
- Potřeba jen jednoho kabelu pro přenos obrazu i zvuku.
- Obraz v maximálním rozlišení (HD) je celkově 2× až 5× podrobnější než obraz ve standardním rozlišení, mezery mezi řádky jsou menší nebo nepostřehnutelné.
- Možnost přenosu až 32 kanálového nekomprimovaného digitálního zvuku.

1.4.5 Nevýhody HDMI

- Konektory jsou „nezátěžové“ a neměly by se kabely ohýbat.
- Kabely s novějšími standardy mají vysokou pořizovací cenu.
- Pro použití vstupu a výstupu zároveň je nutné použít dva samostatné kabely. Jedním kabelem není možný obousměrný přenos dat.

1.5 Internet věcí

V současné době je internet věcí spojován v souvislosti s informačními a telekomunikačními technologiemi. Internet věcí je v současné době možno chápat jako síť propojených zařízení, které mezi sebou komunikují a sdílí si data. Tato komunikace pracuje na standardizovaných komunikačních protokolech. Komunikace se převážně skládá z protokolů TCP/IP. Zařízení je elektronika, která obsahuje software ke komunikaci a sběr dat ze snímačů, které snímají dané veličiny. Tyto zařízení mohou odesílat data pomocí kabelu nebo pomocí bezdrátové technologie.

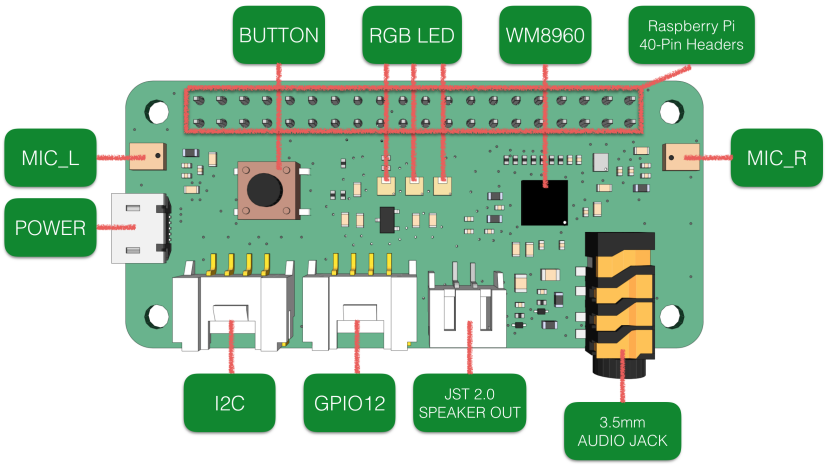
1.6 ReSpeaker 2 Mics Pi HAT

ReSpeaker 2 Mics Pi HAT je rozšiřovací karta s dvěma mikrofony pro Raspberry Pi, která je určena pro AI a hlasové aplikace. Pomocí ní jsme schopni realizovat projekt, protože na desce Raspberry Pi Zero není žádný mikrofon nebo audio výstup. Deska je osazena čipem WM8960 a na stereo kodeku s nízkým výkonem. K dispozici jsou 2 mikrofony na obou stranách desky pro shromažďování zvuků. Dále obsahuje také 3 led diody, jedno tlačítko a pro výstup zvuku je zde zabudován 3,5mm audio jack nebo JST 2.0 Speaker Out. Deska dále obsahuje rozhraní Grove, pomocí kterého se dá rozšířit o další možnosti využití. Deska je napájena pomocí GPIO (General-purpose input/output) sběrnice přímo z Raspberry Pi, nebo se může napájet pomocí microUSB (Universal Serial Bus) konektoru, který je integrován na desce. Pomocí tohoto konektoru se posléze napájí i samotné Raspberry Pi. Celkové rozložení je vyobrazeno na obrázku 1.3. Pro účely bakalářské práce se využívají microfony, tlačítko a audio výstup pomocí 3,5mm audio jacku. [22]

1.7 VoIP

VoIP je technologie, která umožňuje přenos digitalizovaného hlasu pomocí paketů protokolů UDP (User Datagram Protocol), TCP (Transmission Control Protocol) nebo IP (Internet Protocol). Využívá se pro telefonování prostřednictvím Internetu, intranetu nebo jiného datového spojení. K přenosu multimediálních dat využívá

Obr. 1.2: ReSpeaker 2 Mics Pi HAT



Obr. 1.3: Rozložení ReSpeaker 2 Mics Pi HAT

sadu protokolů. Mezi tyto protokoly patří signalizační protokoly (SIP) a protokoly zajišťující přenos multimediálních dat (RTP, SRTP (Secure Real Time Protocol)).

Mezi nejvíce používané signalizační protokoly patří SIP, H.323 nebo IAX (Inter-

Asterisk eXchange). Pro přenos multimediálních dat se používá protokol RTP nebo zabezpečená verze SRTP.

Důležitou součástí VoIP jsou takzvané pobočkové ústředny, koncové zařízení zpravidla VoIP telefony, ale může se v těchto sítích objevit i takzvaná Gateway (brána). Gateway slouží v těchto sítích na propojení IP telefonie a ISDN (Integrated Services Digital Network) sítě. Pobočkové ústředny se starají o registraci uživatelů a zprostředkování hovorů.

Na trhu existuje celá řada výrobců hardwarových ústředen, ale i softwarových ústředen. Poslední součástí jsou koncová zařízení, zpravidla to bývá hardwarový telefon, ale může to být i softwarový telefon (aplikace) nainstalovaný na PC nebo na chytrém mobilním zařízení.

1.8 Detekce VoIP

Důvodem detekce ve VoIP je potřeba internetového poskytovatele vědět, co se všechno přenáší po linkách, tak aby mohl podle toho přizpůsobit síť. Jeden z dalších argumentů proč zachytávat VoIP je snaha o zlepšení zkvalitnění služeb VoIP. Pro zákazníky, kteří si platí tyto služby je důležité, aby poskytnuté služby byly co nejkvalitnější a aby byly spolehlivé.

Jedním z důvodů proč zachytávat VoIP komunikaci je blokace těchto hovorů. Díky rozmachu VoIP komunikace přichází telekomunikační společnosti o své zisky. Firmy hromadně využívají VoIP komunikaci, protože nemusí platiti za tyto služby. [19]

V poslední řadě je tu strach ze zneužití VoIP komunikace k trestné činnosti nebo případně k terorismu a to je důvod proč tyto VoIP hovory detekovat.

1.9 Protokol IP

IP (Internet Protocol) protokol je základním protokolem pracující na síťové vrstvě. Slouží ke komunikaci mezi dvěma počítači. Využívá model TCP/IP ke své komunikaci. Patří do nespojovaných protokolů, to znamená, že před vlastní komunikací není potřeba vytvořit cestu mezi účastníky. IP protokol neumožňuje kontrolu dat ani detekci chyb při přenosu. Technologie VoIP využívá ke svému přenosu protokol IP, ale samotný protokol není nejvhodnější pro komunikaci v reálném čase. [20]

1.10 Protokoly UDP

UDP (User Datagram Protocol) patří do protokolů TCP/IP a je alternativou protokolu TCP. Tento protokol poskytuje nespojitou službu, kde není potřeba před samotnou komunikací vytvořit cestu mezi účastníky. Je zde riziko, že se data po cestě mezi účastníky ztratí. Tento protokol se hodí na použití u technologie VoIP lépe než protokol IP.

1.11 SIP protokol

Session Intiation Protocol (SIP) je signalizační protokol pro sestavení, dohled a rozpad obecných relací (point-to-point nebo multipoint). Je nezávislý na přenášeném obsahu relací. Prostřednictvím protokolu RTP umožňuje přenos multimediálních dat, zpráv a telefonních hovorů.

SIP protokol je poměrně jednoduchý a podobá se HTTP (Hypertext Transfer Protocol) relacím založených na zprávách request/response. Navíc jde o lidsky čitelný text-based formát dat. Podrobnosti o přenášeném obsahu (čísla portů, kodeky, protokoly, šifrování) jsou komunikovány a vyjednávány protokolem SDP (Session Description Protocol).

Koncepce protokolu SIP přenáší komunikační schopnosti na koncová zařízení a sám protokol se soustředí výhradně na řízení komunikace, sestavování, dohled a rozpad relací, což je opačný přístup než mají tradiční telekomunikační sítě, kde koncová zařízení postrádají veškerou inteligenci. Jak název protokolu napovídá, jedná se o protokol relační, ale protože TCP/IP žádnou relační vrstvu nerozeznává, patří SIP mezi protokoly aplikační vrstvy a pracuje nad transportními protokoly UDP (většinou) či TCP. [20]

1.11.1 Adresace v SIP

SIP URI mají různé formy (obecně sip:user@domain) a mohou obsahovat i telefonní čísla.

Podpora jak webové adresace, tak telefonních čísel umožňuje IP komunikaci bez větších problémů přecházet mezi telefonní sítí a Internetem. Uživatelé na kterékoli síti tak mohou komunikovat s kýmkoli na telefonní síti nebo na Internetu, aniž by museli vyměnit svá stávající zařízení. IP zařízení s podporou SIP (telefony, počítače) mohou komunikovat přímo, pokud znají URL (Uniform Resource Locator) druhé strany.

1.11.2 Signalizace

Zprávy protokolu SIP jsou dvojího druhu. A to žádost a odpověď. V tabulce 1.2 jsou nejčastější posílané žádosti v protokolu SIP.

Tab. 1.2: Odesílané žádosti protokolu SIP

Žádost	Popis
REGISTER	registrace účastníka na SIP Proxy serveru
INVITE	zahájení komunikace o plánované nové relaci
ACK	potvrzení zahájení relace
CANCEL	přerušení zahajovaní relace ještě před jejím navázáním
BYE	ukončení probíhající relace
OPTIONS	požádá o informace o možnostech vzdálené strany, aniž by se sestavilo volání

Odpovědi mají tvar trojmístného čísla. Tento návratový kód označuje výsledek žádosti obdobně, jak je tomu například v protokolu HTTP. Lze je rozdělit do šesti skupin podle první číslice. V tabulce 1.3 jsou tyto chyby popsány.

Tab. 1.3: Chybová hlášení protokolu SIP

Číslo chyby	Proces	Popis
1XX	průběh	průběh bez problémů, ale není ukončen přenos
2XX	úspěch	krok ukončen bez problémů
3XX	přesměrování	krok probíhá, ale ještě se v souvislosti s ním něco očekává
4XX	chyba klienta	požadavek je chybný a nemůže být serverem zpracován
5XX	chyba serveru	požadavek je zřejmě v pořádku, ale chyba je na straně serveru
6XX	fatální chyba	zcela fatální chyba, kterou nelze jakkoliv zpracovat

Nejběžnější odpovědi jsou „200 OK“ (úspěšné provedení žádosti), „302 Moved Temporarily“ (běžné přesměrování) a „404 Not Found“ (volaný uživatel se nenachází na daném serveru).

1.11.3 Registrace

Registrace uživatele se provádí odesláním žádosti REGISTER na UAS (User Agent Server). Žádost REGISTER obsahuje v hlavičce protokolu pole TO: kde je obsažena

identita uživatele, respektive SIP URI registrujícího se uživatele. Pole To: společně s polem Contact: vytvoří záznam v registračním serveru, který se následně předán lokalizační službě. Při správné registraci se odesílá potvrzení 200 OK, to znamená, že registrace proběhla úspěšně.

1.11.4 Navázaní a ukončení hovoru

Navázání spojení hovoru se nazývá „three-way handshake“. Klient A, který se chce spojit s klientem B, vyšle požadavek INVITE, kde je obsažen popis nabízeného spojení. Pokud klientovi B přijde zpráva v pořádku, tak odešle zprávu 100. tato odpověď je chápána jako vyzváněcí tón. Když klient B přijme hovor, tak se odešle zpráva 200 OK. Tato zpráva obsahuje IP adresu, čísla portů a kodeky, na kterých bude klient očekávat data. V posledním kroku navazování komunikace odešle klient A klientovi B zprávu ACK, která potvrzuje přijetí odpovědi od klienta B. Po přijetí zprávy ACK začíná přenos audia, popřípadě videa.

Pro ukončení hovoru pošle jeden z účastníků protistraně požadavek BYE, který slouží na ukončení hovoru. Protistrana odpoví potvrzovací zprávou 200 OK a hovor je tímto ukončen.

Spojení obvykle probíhá za účasti SIP serverů, kdy uživatel nemusí znát adresu volaného. Klient A pošle požadavek INVITE, který se dotazuje SIP serveru, ke kterému klient B náleží. SIP server je obvykle spárován s databází kontaktů. Jakmile server zjistí aktuální adresu a polohu volaného klienta, tak tuto adresu posílá zpátky klientu A. Klient A pak rozhodne, zda naváže spojení s klientem B nebo ne. Pokud je SIP server v módu PROXY, tak upraví zprávu INVITE a pošle ji klientovi B. Klient B odešle zprávu 200 OK na SIP server, odkud mu přišla žádost INVITE. Zpráva 200 OK je po přijetí na SIP server přeposílána na klienta A. Po přijetí zprávy 200 OK klientem A se navazuje spojení mezi volanými účastníky.

1.11.5 SIP ústředna

Je zařízení, na kterém je uloženo nastavení zařízení a profily uživatelů. Tyto ústředny mohou být v lokální síti nebo na internetu. Internetové ústředny jsou brány jako služba, kterou poskytuje nějaká společnost. Tyto služby jsou buď volně dostupné a nemusí se za ně platit, nebo jsou placené.

V lokální síti se nejběžněji používá Linuxový server, na který je nahrána SIP ústředna. Nejčastěji se používá SIP ústředna od společnosti Asterisk, která je volně šiřitelná a nejsou za ni účtovány poplatky. Mezi její výhody patří možnost bohatého nastavení a přizpůsobení potřebám daného zákazníka. Dále je tu možnost ukládání výpisů hovorů a monitorování hovorů na svém vlastním hardware. V případě telefonování v lokální síti není potřeba připojení k internetu. Mezi nevýhody tohoto

systému se řadí pořizovací cena hardwaru, nutnost provést vlastní nastavení a zálohovat si systém. Je tu nutnost řešit aktualizace a zabezpečení celého systému.

Na rozdíl od lokálních ústředen jsou internetové ústředny dobře zabezpečené. Uživatel se nemusí starat o zabezpečení a nemusí se také starat o aktualizace systému nebo zálohování systému. Internetoví poskytovatelé nabízejí tyto ústředny řešené přímo na míru zákazníka. Nevýhodou je nutnost stálého připojení k internetu. Bez připojení k internetu nebude fungovat telefonie ani v lokální síti.

1.11.6 Použitá ústředna SIP2SIP.info

SIP2SIP.info je projekt od společnosti AG-Project. Je to komunikační služba v reálném čase pro audio, video, chat, přenos souborů a vícestranné konference založené na signalizaci SIP a souvisejících protokolech (RTP, MSRP (Message Session Relay Protocol) a XCAP (XML Configuration Acces Protocol)). Služba je otevřena a koresponduje s externími doménami SIP a XMPP (Extensible Messaging and Presence Protocol). Celá správa a registrace účtu se provádí přes internetové stránky SIP2SIP.info.

Registrace

Registrace na těchto stránkách je jednoduchá a uživatelsky příjemná. Pro registraci je nutno vyplnit jméno účtu, heslo, e-mail a údaje o uživateli přesněji jeho jméno a příjmení. Po registraci se odešle potvrzovací e-mail na zadanou adresu. Po obdržení potvrzovacího e-mailu se už stačí jen přihlásit k vytvořenému účtu.

Správa účtu

Celé ovládání je prováděno přes webové stránky, které jsou uživatelsky přívětivé a dá se v nich rychle orientovat. Po přihlášení na stránkách SIP2SIP.info se objeví všechny potřebné informace. Dále jsou zde informace o zařízení, na kterém je daný uživatel přihlášen, je tu i možnost náhledu do historie volání, přesměrování hovorů, DNS (Domain Name System), přidávání nebo odebírání kontaktů. Dále se dá nastavit automatický příjem hovorů.

1.12 RTP a SRTP protokol

RTP je síťový protokol pro poskytování zvuku a videa přes IP sítě. Používá se v komunikačních a zábavních systémech, které zahrnují streamování videa, video konference a IP telefonii. V moderní době se nahrazuje zabezpečenou verzí, která má označení SRTP (Secure Real Time Protocol). [20]

1.13 RTCP a SRTCP protokol

RTCP je řídící protokol pro distribuci zvuku a videa v reálném čase. Doplňuje protokoly RTP a SRTP.

RTCP poskytuje řídící informace pro RTP tok dat, ale přitom sám data nenese. Používá se k pravidelnému přenosu kontrolních paketů účastníkům streamované multimediální relace. Hlavní funkcí RTCP je poskytování zpětné vazby na kvalitu služeb poskytovanou RTP. Shromažďuje údaje o multimediálním spojení a informace jako například počet odeslaných paketů, počet ztracených paketů, jitter, zpětnou vazbu a dobu odezvy. Tyto informace může aplikace použít na zvýšení kvality služeb. RTCP neposkytuje šifrování toku dat nebo ověření prostředků. K tomuto účelu může být použit protokol SRTCP. [20]

RTCP stejně jako RTP používá pro přenos dat protokol UDP, ale s rozdílným číslem portu. Definuje pět typů zpráv: Sender Report, Receiver Report, Source Desription Message, Bye Message a Application-Specific Message.

1.14 SDP protokol

SDP je internetový protokol, který je určen k popisu multimediální relace. Nepřenáší se pomocí něj vlastní data, ale slouží pro vyjednání parametrů, jako jsou typ média(audio, video, atd.), transportní protokol (RTP,UDP,IP a atd.), typ kodeku nebo přenosová rychlost.

1.15 Standart H.323

H.323 je standart, který poskytuje multimediální komunikaci v paketových sítích IP. Využívá pro přenos hlasu a videa sadu již existujících protokolů. H.323 je definován standardem ITU-T a poskytuje přenos multimediální komunikace pro různé typy sítí. Doporučuje se používat v sítích , které nepodporují kvalitu služeb. Standart H.323 definuje různé komponenty sítě. Tato síť musí obsahovat koncové zařízení, Gatekeeper, Gateway (brána), MCU (Multipoint Control Unit).

1.15.1 Koncové zařízení

Koncové zařízení je softwarová nebo hardwarová jednotka, která přijímá a vysílá multimediální obsah. Tato jednotka musí podporovat právě standart H.323.

1.15.2 Gatekeeper

Gatekeeper je volitelné zařízení, které se stará o provoz H.323 v počítačové síti. Stará se o překlad adres, řízení přístupu hovorů, kontrola a řízení pásma, řízení zóny, řízení signalizace a automatizaci hovorů.

1.15.3 Gateway (brána)

Gateway slouží ke komunikaci mezi různými typy komunikačních sítí(ISDN (Integrated Services Digital Network),GSM (Global System for Mobile Communications) a atd.). Brána slouží jako koncové zařízení v síti H.323 a sítí kterou propojuje.

1.15.4 MCU (Multipoint Control Unit)

Jednotka MCU se využívá u konferenčních hovorů, kdy se koncová zařízení připojují k této jednotce. MCU jednotka řídí konferenční hovory, kdy vyjednává použité kodeky, které se budou využívat v komunikaci.

1.16 Kodek G.711

Je nejběžnější, nejrozšířenější a nejjednodušší standart pro digitalizaci zvukového signálu. pro svou jednoduchost je však také nejméně úsporný, co se přenosové rychlosti týče. vzorkovací frekvence u tohoto formátu je 8 kHz a rozlišení 8 bitů. Z toho vyplývá přenosová rychlost 64 kbit/s. Tento kodek se používá v ISDN (Integrated Services Digital Network) a PCM (Pulse-code modulation) sítích.

1.17 Kodek G.722

G.722 je širokopásmový zvukový kodek. Tento kodek poskytuje zlepšenou kvalitu řeči díky šířce pásma řeči mezi 50 - 7000 Hz ve srovnání s úzkopásmovým kodekem G.711. Zvuková data se vzorkují frekvencí 16 kHz s rozlišením 14 bitů. Kodek je určen pro aplikace VoIP například v místních sítích, kde je dostupná šířka pásma. G.722 umožňuje přenosovou rychlost 64, 56, 48 kbit/s, ale v praxi je přenosová rychlost 64 kbit/s.

1.18 QOS

QoS (Quality of Service) se používá pro rezervaci a řízení toku dat v telekomunikačních sítích. Protokoly pro QoS se snaží zajistit vyhrazení a dělení dostupné

přenosové kapacity tak, aby nedocházelo k zahlcení sítě a ke snížení kvality služeb. Technologie QoS se využívá hlavně v sítích, kde se pracuje s VoIP nebo s přenosem multimediálních dat jako je IPTV (Internet Protocol television). QoS pracuje tak, že upřednostňuje některé služby nad ostatníma. Toto upřednostňování se dá nastavit. Hlavně se upřednostňují služby, které potřebují, aby zpoždění bylo co nejmenší. [21]

1.19 WI-FI

Technologie WIFI je bezdrátová komunikace založena na standardech IEEE 802.11, které popisují bezdrátovou komunikaci v počítačových sítích. WIFI využívá tak zvaného bezlicenčního pásma, proto je ideální na budování levné počítačové sítě bez nutnosti pokládání kabelů. Samotná bezdrátová technologie byla publikována v roce 1997 organizací IEEE (Institute of Elestrical and Electronics Engineers). Samotný pojem WIFI vznikl v roce 1999. Bezdrátová komunikace probíhá na pásmech 2,4 a 5 GHz. V současné době existuje několik standardú 802.11, které se využívají. [18]

1.19.1 802.11

Tento standart vznikl v roce 1997 a často bývá označen jako 802.11 legacy. 802.11a bylo představeno v roce 1999. Standart se už téměř nikde nevyužívá a byl nahrazen novějšími standarty. Maximální přenosová rychlost tohoto standartu je 2 Mb/s a pracuje na frekvenci rádiových vln 2,4 GHz.

1.19.2 802.11a

Standart 802.11a oproti standartu 802.11 legacy pracuje v pásmu 5GHz, které má své výhody i nevýhody. Tento standart dosahoval maximální přenosové rychlosti 54 Mb/s.

1.19.3 802.11b

Standart 802.11b pracuje na přenosovém pásmu 2,4 GHz. Maximální přenosová rychlost tohoto standartu je 11 Mb/s. Byl představen v roce 1999 spolu se standardem 802.11a. ů

1.19.4 802.11g

802.11g byl ratifikován v roce 2003, tento standart stejně jako standard 802.11b pracuje na přenosovém pásmu 2,4 GHz. Maximální přenosová rychlost tohoto standartu je 54 Mb/s.

1.19.5 802.11n

Velkou změnou byl standart 802.11n, který pracuje v pásmu 2,4 a 5 Ghz. Tento standart byl publikován v roce 2009. Maximální přenosová rychlost, které tento standart může dosáhnout je 600 Mb/s. Velkým přínosem tohoto standartu je využití více antén pro přenos. U tohoto standartu se objevila metoda MIMO (Multiple-input multiple-output). U standartu 802.11n se objevila možnost nastavení nastavit šířku přenášeného pásma. Lze tedy využít šířku přenášeného pásma 20 MHz a 40 MHz.

1.19.6 802.11ac

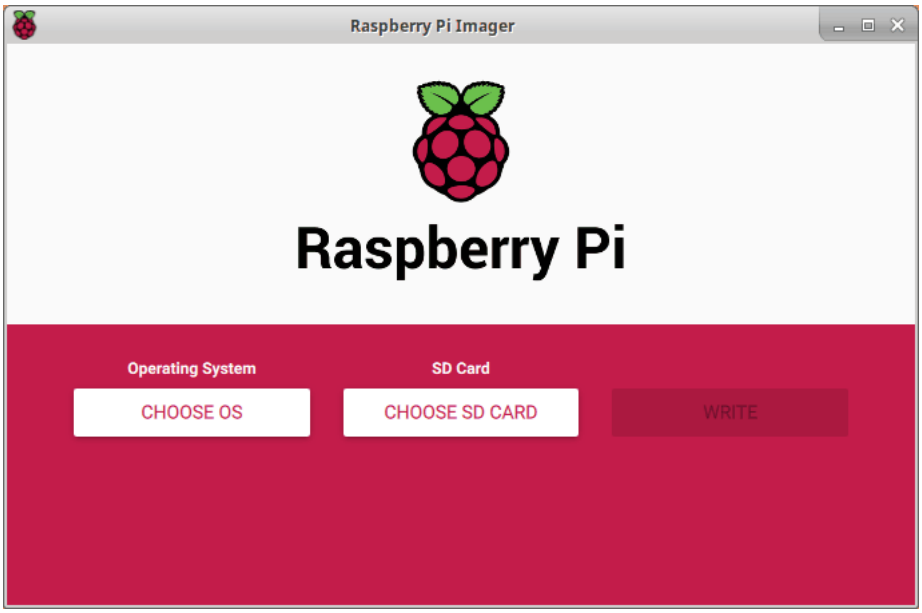
Standard 802.11ac byl schválen v roce 2014. Stejně jako standard 802.11n pracuje v pásmu 2,4 a 5 GHz. U tohoto standardu byla přidána možnost využít šířku přenášeného pásma 80 MHz a 160 MHz. Maximální přenosová rychlost u tohoto standartu je 1Gb/s.

2 Návrh obslužného programu

Tato část bakalářské práce je zaměřená na vytvoření funkčního projektu. Kapitola je rozdělena do částí podle toho jak se projekt realizoval. První část této kapitoly je zaměřena na instalaci potřebného operačního systému na paměťovou kartu. Druhá část této kapitoly se zabývá instalací a konfigurací rozšiřujícího modulu Respeaker 2 Mics Pi HAT. Předposlední část této kapitoly je zaměřena na stažení, kompilaci a ověření funkčnosti PJSIP. Poslední část kapitoly se zabývá vytvořením skriptu na ověření funkčnosti tlačítka. A následně vytvořením skriptů na ovládání VoIP. Skripty na ovládání VoIP jsou napsány v programovacím jazyce Python.

2.1 Instalace operačního systému

Pro samotnou instalaci operačního systému je nutné mít RPi, SD kartu a počítač s připojením na internet, odkud se stáhne nejnovější verze Raspbianu. Pomocí počítače se stáhne ze stránek výrobce instalační soubor a spustí se instalace. Na obrázku 2.1 je vyobrazena aplikace Raspberry Pi Imager, pomocí které se zapíše nejnovější operační systém na paměťovou kartu. Nejdříve se musí vybrat verze operačního systému, to se provede tím, že se klikne na CHOOSE OS a vybere se Raspberry Pi OS (others). Po kliknutí se zobrazí nabídka, ze které se vybere se Raspberry Pi OS Lite (32-bit). Tato verze operačního systému je bez grafického rozhraní a další nastavení se provádí pomocí příkazového řádku.



Obr. 2.1: Raspberry Pi Imager

Verze operačního systému bez grafického rozhraní je volena z důvodu, že zvolené RPi nebude připojeno k žádnému zobrazovacímu médiu. Dále se musí vybrat paměťová karta, kam se zvolený operační systém nainstaluje. Nakonec se zmáčkne tlačítko WRITE, které nainstaluje daný operační systém na paměťovou kartu. Po skončení tohoto procesu se paměťová karta z počítače ještě nevyjme, protože je nutné na ni vložit ještě dva soubory. Tyto soubory se vloží na disk boot. První z těchto souborů se jmenuje *ssh* a nemá žádnou příponu. Tento soubor slouží k povolení vzdáleného připojení pomocí ssh na RPi.

Druhý soubor je jmenuje *wpa_supplicant.conf* a slouží k nastavení připojení RPi k bezdrátové síti. Jeho struktura je vypsaná níže:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=CZ

network={
  ssid="NAZEV-SSID"
  psk="HESLO"
  key_mgmt=WPA-PSK
}
```

Po uložení těchto souborů do disku boot se paměťová karta vyjme z počítače a vloží se do RPi. Po vložení paměťové karty do RPi se připojí napájení. Jelikož k RPi není připojeno žádné zobrazovací médium, tak se na RPi bude připojovat vzdáleně pomocí programu Putty. Před připojením k RPi je nutné si zjistit IP adresu zařízení. To se provede pomocí programu Angry IP Scanner, který slouží ke skenování připojených zařízení do dané počítačové sítě. Po zjištění IP adresy RPi je nutné spustit program Putty a do něj zadat IP adresu RPi a SSH port. SSH pracuje v základním nastavení na portu 22, který se zadá do kolonky Port. Po zadání příslušných údajů stačí jen zmáčknout tlačítko Open a počkat na sestavení spojení mezi PC a RPi. Při sestavování spojení vyskočí tabulka s informacemi o RSA (public-key cryptosystem) klíči a je nutné kliknout na „Accept a save“. Toto potvrzení způsobí, že při příštím přihlašování se už nebude muset ověřovat RSA klíč.

Pro přihlášení je nutné zadat uživatelské jméno a heslo. Uživatelské jméno je nastaveno v základu na: „pi“ a heslo na: „raspberry“. Po přihlášení je nutné zadat tyto příkazy:

```
$ sudo apt-get update
$ sudo apt-get upgrade -y
```

Příkazy, které jsou zde uvedené zapříčiní, že systém se aktualizuje a nainstalují se nejnovější balíčky. Po provedení těchto příkazů je systém připraven na další krok.

Může být proveden ještě jeden příkaz, pomocí kterého se může nastavit uživatelské jméno, heslo, název zařízení v síti a další parametry.

```
$ sudo raspi-config
```

2.2 Instalace rozšiřujícího modulu ReSpeaker 2 Mics Pi HAT

Rozšiřující modul Respeaker 2 Mics Pi Hat se připojuje k RPi pomocí 40-pinového konektoru GPIO. Přes tento konektor se rozšiřující modul napájí a probíhá přes něj samotná komunikace. Tato podkapitola je rozdělena do dalších částí podle toho jak se rozšiřující modul zprovozňoval. První část se zabývá přípravou RPi před samotnou instalací rozšiřujícího modulu. V další části je popsána samotná instalace ReSpeakeru a ověřením jeho funkčnosti. Poslední část této podkapitoly se zabývá úpravou spouštěcího souboru.

2.2.1 Příprava Raspberry Pi

Tento krok se provede jen tehdy, pokud je ReSpeaker instalován na již používaný RPi. Před samotnou instalací je nutné provést dva příkazy, které aktualizují RPi. A to jsou tyto příkazy:

```
$ sudo apt-get update
$ sudo apt-get upgrade -y
```

První příkaz aktualizuje všechny používané repozitáře. Druhým příkazem se nainstalují nejnovější verze používaných aplikací a systém je připraven na další krok a to je instalace potřebného softwaru pro ReSpeaker.

2.2.2 Instalace softwaru ReSpeaker

Instalaci softwaru pro ReSpeaker vyžaduje nainstalovat Git, který umožní stahovat soubory z GitHabu. Instalace Gitu se provede tímto příkazem:

```
$ sudo apt-get install git -y
```

Samotná instalace je rozdělena do více kroků. V prvním kroku se stáhne z GitHabu seed-voicecard, kde jsou obsaženy všechny potřebné soubory pro samotnou instalaci. Stažení se provede tímto příkazem:

```
$ git clone https://github.com/respeaker/seed-voicecard.git
```

Po stažení je nutné si otevřít složku seed-voicecard a to je provedeno tímto příkazem:

```
$ cd seed-voicecard
```

Po otevření složky je nutné zadat příkaz na instalaci všech potřebných podprogramů. Instalace se provede tímto příkazem:

```
$ sudo ./install.sh
```

Po provedení instalace je nutné provést restart systému a to se provede tímto příkazem:

```
$ sudo reboot
```

Po restartu je nutné se znovu přihlásit do systému. Následným příkazem se ověří, zda připojenou zvukovou kartu systém vidí. Toto se provede příkazem:

```
$ aplay -l
```

Tento příkaz vypíše všechny připojené zvukové karty. Následným příkazem se ověří viditelnost všech záznamových zařízení.

```
$ arecord -l
```

Tento příkaz vypíše seznam všech připojených zařízení, které mají schopnost zaznamenat zvuk. Pokud systém vidí připojenou zvukovou kartu ReSpeaker, tak je tento krok poslední,co se instalace týče. Poté už je pouze upraven spouštěcí soubor.

2.2.3 Úprava spouštěcího souboru

Posledním krokem v instalaci rozšiřující karty je úprava spouštěcího souboru. Tato úprava spouštěcího souboru způsobí, že systém bude při startu používat rozšiřující modul ReSpeaker 2 Mics Pi HAT pro přehrávání zvuku a jeho zachycení. K tomuto kroku je nutné provést úpravu souboru, který se nachází na disku boot. Úprava souboru se provede tímto příkazem:

```
$ sudo nano /boot/config.txt
```

V tomto souboru se pouze upraví řádek, který odkazuje na *dtparam=audio=on*, tento řádek se nachází v dolní části souboru. Koncový soubor *config.txt* po úpravě bude vypadat takto:

```
# Enable audio (loads snd_bcm2835)
dtparam=audio=off
dtoverlay=i2s-mmap
dtoverlay=seed-2mic-voicecard
```

Po změně tohoto souboru je nutné provést restart systému, aby se zapsaly všechny změny a systém se spustil s již upraveným nastavením. Restart systému se provede následujícím příkazem:

```
$ sudo reboot
```

Po restartu je nutné ověřit, zda se provedené změny zapsaly správně. Následujícím příkazem se znovu vypíše seznam zařízení na přehrávání zvuku s tím rozdílem, že po provedené změně se v tomto seznamu objeví jen jedna zvuková karta.

```
$ aplay -l
```

Posledním příkazem se vypíše seznam všech zařízení na zaznamenávání zvuku. Pokud se úprava souboru *config.txt* provedla správně, tak se vypíše jen jedno zařízeni.

```
$ arecord -l
```

Tímto je zvuková karta ReSpeaker 2 Mics Pi Hat připravena na používaní.

2.3 PJSIP

Tato podkapitola se zabývá popisem PJSIP a následného stažení a implementaci na RPi. PJSIP je volně dostupná a open source multimediální knihovna, která implementuje základní protokoly pro multimediální komunikaci. Je založena na protokolech SIP, RTP a SDP. Kombinuje signalizační protokol SIP s funkcemi NAT (Network Address Translation) a vytváří tak vysoce kvalitní multimediální komunikační API (Application Programing Interface), které se dá využít na jakémkoliv systému.

2.3.1 Příprava systému na stažení PJSIP

Tento krok se provede jen tehdy, pokud se bude PJSIP stahovat a kompilovat na již používané RPi. Před samotným stahováním PJSIP je nutné systém RPi aktualizovat a to se provede následujícími příkazy:

```
$ sudo apt-get update
$ sudo apt-get upgrade -y
```

Po provedení těchto příkazů se systém aktualizuje a následně se může přejít na samotné stažení PJSIP na RPi.

2.3.2 Stažení a kompilace PJSIP

Před samotným stažením PJSIP je nutné na RPi nainstalovat Git. Pokud tento nástroj už používáte, tak se tento krok přeskočí a je možné stáhnout PJSIP.

Nástroj Git se nainstaluje příkazem:

```
$ sudo apt-get install git -y
```

Pro pozdější správné spuštění PJSIP je nutné doinstalovat potřebné balíčky. Instalace všech potřebných balíčků se provede tímto příkazem:

```
$ sudo apt-get install libssl-dev libasound2-dev \
libasound2-doc libasound2-plugins libasound2 \
libasound-dev build-essential automake autoconf \
libtool libpulse-dev libsamplerate0-dev \
libcommoncpp2-dev libccrtp-dev libzrtpcpp-dev \
libdbus-1-dev libasound2-dev libdbus-c++-dev libyaml-dev \
libpcre3-dev libgsm1-dev libcelt-dev alsa-base alsa-utils \
alsa-tools opus-tools libopus-dev libpcre3-dev \
libcommoncpp2-dev libdbus-1-dev libyaml-dev libv4l-dev \
libvo-amrwbenc-dev -y
```

Po této instalaci je nutné na GitHabu stáhnout nejnovější verzi PJSIP. Při tvorbě této práce byl využit PJSIP ve verzi 2.10. V této verzi byly provedeny úpravy skriptů na kompilaci. Stažení PJSIP se provede pomocí tohoto příkazu:

```
$ wget https://github.com/pjsip/pjproject/archive/2.10.tar.gz
```

Po stažení PJSIP je nutné provést rozbalení tohoto souboru příkazem:

```
$ tar xfv 2.10.tar.gz
```

Nyní je potřeba otevřít vytvořenou složku *pjproject-2.10* to se provede následujícím příkazem.

```
$ cd pjproject-2.10
```

Po otevření složky je nutné provést úpravu souboru *config_site.h*, který se nachází v podsložce *pjlib/include/pj*. Úprava souboru se provede tímto příkazem:

```
$ nano config_site.h
```

Po otevření tohoto souboru je nutné ho upravit.

```
#define PJMEDIA_AUDIO_DEV_HAS_ALSA      1
#define PJMEDIA_AUDIO_DEV_HAS_PORTAUDIO 0
#define PJMEDIA_HAS_VIDEO       0
```

Dále je nutné upravit soubor *user.mak*, který se nachází v kořenovém adresáři *pjproject-2.10*. Do tohoto adresáře se vrátíme pomoci příkazu:

```
$ cd ~/pjproject-2.10
```

Úprava souboru *user.mak* se provede následujícím příkazem:

```
$ nano user.mak
```

Pro použití PJSIP na RPi je nutné upravit tento soubor tak, aby vypadal podle tohoto vzoru:

```
# You can create user.mak file in PJ root directory to specify
# additional flags to compiler and linker. For example:
export CFLAGS += -fPIC -mcpu=arm1176jzf-s -mfpu=vfp
-ffast-math -mfloat-abi=hard -mlittle-endian -munaligned-access
-DPJ_HAS_IPV6=1
export LDFLAGS +=
```

Po úpravě těchto souborů je nutné provést tento příkaz:

```
./configure --enable-shared \
--disable-video \
--disable-l16-codec \
--disable-speex-aec \
--disable-speex-codec \
--disable-ilbc-codec \
--disable-sdl \
--disable-ffmpeg \
--disable-v4l2 \
--disable-openh264 \
--disable-libwebrtc \
--disable-floating-point \
--disable-large-filter \
--disable-gsm-codec \
--disable-g722-codec \
--disable-g7221-codec
```

Po provedení tohoto příkazu je nutné rozšířit swappovací prostor. Tento prostor je možné rozšířit pomocí těchto příkazů:

```
$ sudo swapoff
$ sudo dd if=/dev/zero of=/tempswaú bs=1024 count=1000k
$ sudo mkswap /tempswap
$ sudo swapon /tempswap
$ free -m
```

Příkaz *free -m* vypíše velikost volného místa v paměti. Kdyby se swapovací prostor nerozšířil, tak by se kompilace neprovedla správně a musela by se swapovací

paměť dodatečně rozšířit. Před samotnou kompilací je nutné se dostat do podadresáře */pjsip-apps/src/swig*. Tento podadresář se otevře tímto příkazem:

```
$ cd pjsip-apps/src/swig
```

Po otevření podadresáře se musí spustit dva příkazy, které provedou kompilaci PJSIP.

```
$ make
$ sudo make install
```

Příkaz *make* je časově náročný a vyžaduje větší swappovací prostor. Po provedení příkazů je kompilace dokončena. Pro ověření funkčnosti stačí zadat následující příkaz, který vytvoří testovací hovor. Pokud se kompilace provedla správně, tak testovací hovor naváže spojení a v zápětí se ukončí.

```
$ python3 python/test.py
```

Aby vše fungovalo správně je nutné upravit soubor *daemon.conf*, který se nachází v adresáři */etc/pulse*. Tento soubor je nutné upravit podle tohoto vzoru, který se nachází v příloze A.3 na straně 52

Lze provést ještě jeden test, zda byla kompilace úspěšná. Je nutné se přepnout do složky *pjsip-apps/bin*. Do této složky se dá přepnout pomocí následujícího příkazu.

```
$ cd ~/pjproject-2.10/pjsip-apps/bin
```

Test se provede pomocí příkazu, který spustí aplikaci PJSIP. V následujícím příkazu je nutno nahradit *XXXX* přihlašovacími údaji z použité ústředny SIP2SIP.info.

```
./pjsua-armv6l-unknown-linux-gnueabihf --id sip:XXXX
--registrar sip:XXXX --realm sip2sip.info --username XXXX
--password XXXX --contact sip:XXXX
--outbound sip:proxy.sipthor.net:5060 --thread-cnt 1
--nameserver 8.8.8.8 --nameserver 8.8.4.4 --publish
--clock-rate 8000 --add-codec pcma --dis-codec opus/48000/2
--dis-codec AMR-WB/16000/1 --dis-codec AMR/8000/1
```

2.4 Návrh obslužného programu

Poslední podkapitola této kapitoly se zabývá vytvořením skriptů, které slouží na ovládání celého zařízení RPi. Návrh obslužného programu je rozdělen do více část. V první části se řeší zprovoznění tlačítka, které je zabudované na rozšiřující desce ReSpeakr 2 Misc Pi HAT, které je připojeno k RPi pomocí sběrnice GPIO.

Další část této podkapitoly se zabývá vytvořením samotných skriptů, které slouží k ovládání PJSIP.

2.4.1 Instalace a ověření funkčnosti tlačítka

Hardwarové tlačítko se nachází na rozšiřující desce ReSpeaker 2 Mics Pi HAT, která je připojena pomocí 40-pinového konektoru GPIO. Tlačítko je připojeno k této sběrnici na pinu 17. Na ověření funkčnosti tlačítka je nutné doinstalovat balíček k programovacímu jazyku Python. Instalace tohoto balíčku se provede následujícími příkazy.

```
$ sudo apt-get install python-pip -y
$ sudo pip install rpi.gpio -y
```

Po nainstalování balíčku *rpi.gpio* je nutné vytvořit jednoduchý skript, který bude vypisovat do konzole informaci o aktivnosti stisknutého tlačítka. Na internetu je více skriptů, které jsou volně dostupné a slouží ke stejnému nebo podobnému účelu. Zde je ukázka tohoto skriptu:

```
import RPi.GPIO as GPIO
import time

BUTTON = 17

GPIO.setmode(GPIO.BCM)
GPIO.setup(BUTTON, GPIO.IN)

while True:
    state = GPIO.input(BUTTON)
    if state:
        print("off")
    else:
        print("on")
time.sleep(1)
```

Na začátku je nutné importovat knihovny, které se využívají. Následně je zde nadeklarované tlačítko s přiřazeným pinem sběrnice GPIO. V dalším kroku je nutné nastavit sběrnici GPIO. v *GPIO.BCM* se nastavuje číslování pinů podle Broadcom čipu. Lze místo *BCM* využít i *BOARD*, které nastavuje číslování pinů v kruhu. Doporučuje se místo *BOARD* využívat *BCM*, kvůli přehlednosti výstupních pinů. V dalším kroku je nutné nastavit tlačítko tak, aby bylo vstupním elementem. V poslední části kódu je vytvořen while cyklus (nekonečná smyčka), který testuje zda je tlačítko rozepnuté nebo sepnuté.

Vytvořený kód se spustí pomocí následujícího příkazu:

```
$ python tlacitko.py
```

Po spuštění se do terminálu začne vypisovat zda je tlačítko sepnuté nebo rozepnuté. V skriptu je nastavena nekonečná smyčka, takže ukončení tohoto programu se provede zmáčknutím kombinace kláves *Ctrl + c*, tím dojde k přerušení skriptu.

2.4.2 Tvorba skriptů

Tato sekce se zabývá vytvořením skriptů. Celkem jsou vytvořeny tři skripty, které mají různé úkoly. První skript je spouštěcí skript. Který se spustí s přihlášením uživatele do operačního systému. Tento skript je pojmenován *runscript.sh* a je v příloze A.5 na straně 55. Aby se spustil skript *runscript.sh*, tak je nutné provést úpravu souboru *rc.local*. To se provede následujícím způsobem:

```
$ sudo nano /etc/rc.local
```

Je nutné se v tomto souboru dostat nakonec a před *exit 0* zadat „/boot/runscript.sh &“. Tímto se dosáhne spuštění požadovaného scriptu po spuštění systému.

Prvním úkolem tohoto skriptu je překopírování souboru *wpa_supplicant.conf* z adresáře *etc/wpa_supplicant* do disku *boot*. Tento skript je v příloze A.6 na straně 56. Posledním úkolem skriptu *runscript.sh* je spuštění skriptu *call.py*. Zde se provede registrace účastníka a následně se může vytvořit hovor, nebo přijmout příchozí hovor. Dále dokáže zachytávat dtmf (čísla zadaná na telefonu) a tyto čísla vypíše následně do konzole. Tento skript se nachází v příloze A.7 na straně 57.

Závěr

Tato bakalářská práce měla za cíl sestrojit jednoduchý VoIP komunikátor. Při tvorbě bylo použito zařízení RPi Zero WH, ke kterému byl připojen rozšiřující modul ReSpeaker 2 Mics Pi HAT. Jako operační systém byl využit systém nabízený společnosti Raspberry Pi Foundtion v nejnovější verzi s označením Buster. Instalace a nastavení operačního systému proběhlo bez problému. Po prvním spuštění se zjistilo, že vytvořený soubor (*wpa_supplicant.conf* na disku *boot* se přesunul do složky */etc/wpa_supplicant*. Tímto přesunutím bylo znemožněno následné úpravy souboru, tudíž byl vytvořen skript, který tento soubor zkopíroval zpátky na disk*boot* a tím umožnil následné úpravy tohoto souboru.

Při instalaci rozšiřující karty ReSpeakr 2 Mics Pi HAT bylo zjištěno, že nejnovější operační sytém využívá jádro systému ve verzi 4.19 a zvuková karta má podporu pro jádro systému 3.18. Ale s tímto nebyl problém, protože při instalaci potřebných balíčků se z internetu stáhly všechny potřebné soubory, které umožňují práci této zvukové karty na jádru systému 4.19 a novějších verzí. Nejnovější verze jádra systému je 5.4. Instalace potřebných balíčků pro správnou funkci zvukové karty trvala přibližně 10 minut. Po instalaci byla ověřena funkčnost zvukové karty.

Po otestování zvukové karty probíhala samotné stažení PJSIP a následná kompilace. Před samotnou kompilací bylo potřeba stáhnou a nainstalovat potřebné balíčty, tak aby PJSIP fungoval správně. Po nainstalování balíčků bylo nutno stáhnout nejnovější verzi PJSIP a upravit konfigurační balíčky. Při kontrole konfiguračních balíčků bylo zjištěno, že nejnovější verze 2.10 má opravené konfigurační balíčky a tudíž nebyla potřeba je upravovat oproti verzi 2.9. Byly provedeny dvě kompilace. První kompilace byla bez problémů, ale při druhé kompilaci se vyskytly problémy. Tyto problémy byly způsobeny malým swapovacím oddílem, tudíž bylo nutno tento prostor rozšířit. Po rozšíření prostoru byla provedena druhá kompilace, která trvala skoro půl hodiny. Tento čas byl dlouhý z toho důvodu, že RPi Zero má jednojádrový procesor s nízkým taktem. Po provedení obou kompilací byla vyzkoušena funkčnost PJSIP. Při ověřování funkčnosti byla zjištěn problém se zvukem, ale ten byl následně opraven. Byla potřeba opravit soubor *daemon.conf*. Po opravě tohoto souboru nebyly zjištěny další problémy. Po kompilaci a otestování PJSIP přišly na řadu spouštěcí skripty. První skript, který byl vytvořen byl skript, který kopíruje soubor *wpa_supplicant.conf*. Při tvorbě tohoto skriptu se projevila nedostačující znalost programovacího jazyku Python. Kopírovací skript bal vytvořen a odzkoušen zda správně funguje. Posléze byl vytvořen hlavní spouštěcí skript, který na začátku volal jen kopírovací skript. Nakonec byl vytvořen skript, který má na starosti zaregistrování účastníka a vytvoření hovoru, čekání na příchozí hovor a výpis dtmf kódu do konzole. S tímto skriptem byly problémy, kvůli pochopení činnosti předvytvořených

skriptů a následné implementace všech potřebných součástí. Při tvorbě byla využity návody a popisy všech funkcí na stránkách výrobce. Tyto stránky byly nepřehledné a některé funkčnosti nebyly vůbec popsány. Tvorba tohoto skriptu je velmi náročná i pro zkušené programátory. Skript byl vytvořen, ale nebyl odzkoušen kvůli časové náročnosti projektu. Všechny vytvořené skripty jsou vystaveny na GitHabu pod licencí MIT. Odkaz je zde: `https://github.com/Devil811/VoIP-communicator`.

Možné využití této bakalářské práce je jako inteligentní domovní zvonek, který je zabudovaný v krabičce s reproduktorem a s relé, které otevře dveře při zadání určité kombinace dtmf kódu.

Literatura

[1] Linux Dokumentační projekt. 4. vyd. Computer Press, 2008. 1336 s. ISBN: 978-80-251-1525-1.

[2] PILGRIM, M. Ponořme se do Python(u) 3. CZ.NIC, 2010. 435 s. ISBN: 978-80-904248-2-1.

[3] Raspberry Pi. Raspberry [online]. [cit. 2020-06-08]. Dostupné z: https://www.raspberrypi.org/

[4] Raspberry Pi Zero. Raspberry Pi Zero [online]. [cit. 2020-06-08]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-zero/

[5] Raspberry Pi 3. Raspberry Pi 3 [online]. [cit. 2020-06-08]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[6] Raspberry Pi 4. Raspberry Pi 4 [online]. [cit. 2020-06-08]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

[7] Ubuntu. Ubuntu Mate [online]. [cit. 2020-06-08]. Dostupné z: https://ubuntu-mate.org/ports/raspberry-pi/

[8] OSMC. OSMC [online]. [cit. 2020-06-08]. Dostupné z: https://osmc.tv/

[9] LibreELEC. LibreELEC [online]. [cit. 2020-06-08]. Dostupné z: https://libreelec.tv/

[10] PiNet. PiNet [online]. [cit. 2020-06-08]. Dostupné z: http://pinet.org.uk/

[11] Windows pro Internet věcí. Windows [online]. [cit. 2020-06-08]. Dostupné z: https://developer.microsoft.com/cs-cz/windows/iot/

[12] Arch Linux. Arch Linux [online]. [cit. 2020-06-08]. Dostupné z: https://www.archlinux.org/

[13] Stručný popis sběrnice I2C a její praktické využití k připojení externí eeprom 24LC256 k mikrokontroléru PIC16F877. Vyvoj.hw.cz [online]. [cit. 2020-06-08]. Dostupné z: https://vyvoj.hw.cz/navrh-obvodu/strucny-popis-sbernice-i2c-a-jeji-prakticke-vyuziti-k-pripojeni-externi-eeprom-24lc256

[14] Externí sériové sběrnice SPI a I²C. Root.cz [online]. [cit. 2020-06-08]. Dostupné z: https://www.root.cz/clanky/externi-seriove-sbernice-spi-a-i2c/

[15] JANČO, Tomáš. ANALYZÁTOR KOMUNIKACE NA SBĚRNICI USB. Brno, 2016. Diplomová práce. VUT Fakulta informačních technologií. Vedoucí práce Ing. Václav Šimek.

[16] BORTLOVÁ, Pavlína. SOUBOROVÉ SYSTÉMY NA RŮZNÝCH TYPECH PAMĚŤOVÝCH MÉDIÍ. Brno, 2016. Bakalářská práce. VUT Fakulta informačních technologií. Vedoucí práce Ing. Jakub Lojda.

[17] MAREK, Jan. VYUŽITÍ ROZHRANÍ HDMI NA VÝUKOVÉM KITU MINERVA. Brno, 2017. Bakalářská práce. VUT Fakulta informačních technologií. Vedoucí práce Ing. Václav Šimek.

[18] KRULICH, Filip. POKROČILÝ ROAMING VE WI-FI SÍTÍCH. Brno, 2019. Bakalářská práce. VUT Fakulta elektrotechniky a komunikačních technologií. Vedoucí práce Ing. Petr Ilgner.

[19] HAVELKA, Ondřej. DETEKCE VOIP APLIKACÍ VE STATISTIKÁCH SÍŤOVÉHO PROVOZU. Brno, 2009. Bakalářská práce. VUT Fakulta informačních technologií. Vedoucí práce Ing. Martin Žádník.

[20] BĚLÍK, David. PROPRIETÁRNÍ VOIP PROTOKOLY VÝROBCŮ POBOČKOVÝCH ÚSTŘEDEN. Brno, 2011. Bakalářská práce. VUT Fakulta informačních technologií. Vedoucí práce Ing. Pavel Šilhavý, Ph.D.

[21] DANKO, Martin. MODELOVÁNÍ KVALITY SLUŽEB V POČÍTAČOVÝCH SÍTÍCH. Brno, 2012. Diplomová práce. VUT Fakulta informačních technologií. Vedoucí práce Ing. Ondřej Ryšavý, Ph.D.

[22] ReSpeaker 2 Mics Pi HAT. RPishop.cz [online]. [cit. 2020-06-08]. Dostupné z: https://rpishop.cz/zvukove-karty/1173-respeaker-2-mics-pi-hat.html

Seznam symbolů, veličin a zkratek

MSRP Message Session Relay Protocol

XCAP XML Configuration Acces Protocol

XMPP Extensible Messaging and Presence Protocol

ISDN Integrated Services Digital Network

GSM Global System for Mobile Communications

PCM Pulse-code modulation

SD karta paměťové zařízení

SSH Secure Shell

ARM Ashton Raggatt McDougall

GPIO General-purpose input/output

USB Universal Serial Bus

SRTP Secure Real-time Transport Protocol

HTTP Hypertext Transfer Protocol

URL Uniform Resource Locator

IAX Inter-Asterisk eXchange

SRTP Secure Real Time Protocol

IP Internet Protocol

TCP Transmission Control Protocol

UDP User Datagram Protocol

H.323 kodek

G.711 kodek

G.722 kodek

open source Otevřený software

UAC User Agent Clien

UAS	User Agent Server
DHCP	Dynamic Host Configuration Protokol
DNS	Domain Name System
Pi-Hole	Blokování internetové inzerce
RPi	Raspberry Pi
VoIP	Voice over Internet Protocol
SIP	Session Initiation Protocol
RTP	Real-time Transport Protocol
RTCP	RTP Control Protocol
SRTCP	Secure RTP Control Protocol
SDP	Session Description Protocol
RSA	public-key cryptosystem
API	Application Programing Interface
NAT	Network Address Translation
HDMI	High-Definition Multimedia Interface
WIFI	
MIMO	Multiple-input multiple-output
IEEE	Institute of Elestrical and Electronics Engineers
QoS	Quality of Service
IPTV	Internet Protocol television
HD	High-Definition
HDR	High Dynamic Range Imaging
3D	Three-dimensional space
DVI	Digital Visual Interface
DVD	Digital Versatile Disc

YCbCr

RGB	Reg-Green-Blue
CEC	Consumer Electronics Control
I2C	Inter Integrated Circuit
SDA	Serial Data Line
SCL	Serial Clock Line
SPI	Serial Peripheral Interface
SCK	Serial Clock
MOSI	Master Out, Slave In
MISO	Master In, Slave Out
SS	Slave Select
OEM	Original Equipment Manufacture

Seznam příloh

A Zdrojové kódy

Výpis A.1: Soubor config_site.h

```
#define PJMEDIA_AUDIO_DEV_HAS_ALSA      1
#define PJMEDIA_AUDIO_DEV_HAS_PORTAUDIO 0
#define PJMEDIA_HAS_VIDEO       0
```

Výpis A.2: Soubor user.mak

```
export CFLAGS += -fPIC -mcpu=arm1176jzf-s -mfpu=vfp
-ffast-math -mfloat-abi=hard -mlittle-endian
-munaligned-access -DPJ_HAS_IPV6=1
export LDFLAGS +=
```

Výpis A.3: Soubor deamon.conf

```
; daemonize = no
; fail = yes
; allow-module-loading = yes
; allow-exit = yes
; use-pid-file = yes
; system-instance = no
; local-server-type = user
; enable-shm = yes
; enable-memfd = yes
; shm-size-bytes = 0
; lock-memory = no
; cpu-limit = no

; high-priority = yes
; nice-level = -11

; realtime-scheduling = yes
; realtime-priority = 5

; exit-idle-time = 20
; scache-idle-time = 20

; dl-search-path = (depends on architecture)

; load-default-script-file = yes
; default-script-file = /etc/pulse/default.pa

; log-target = auto
; log-level = notice
; log-meta = no
; log-time = no
; log-backtrace = 0

; resample-method = speex-float-1
; enable-remixing = yes
; enable-lfe-remixing = no
; lfe-crossover-freq = 0
```

```
; flat-volumes = yes

; rlimit-fsize = -1
; rlimit-data = -1
; rlimit-stack = -1
; rlimit-core = -1
; rlimit-as = -1
; rlimit-rss = -1
; rlimit-nproc = -1
; rlimit-nofile = 256
; rlimit-memlock = -1
; rlimit-locks = -1
; rlimit-sigpending = -1
; rlimit-msgqueue = -1
; rlimit-nice = 31
; rlimit-rtprio = 9
; rlimit-rttime = 200000

; default-sample-format = s16le
; alternate-sample-rate = 48000
; default-sample-channels = 2
; default-channel-map = front-left,front-right

; default-fragments = 4
; default-fragment-size-msec = 25

; enable-deferred-volume = yes
; deferred-volume-safety-margin-usec = 8000
; deferred-volume-extra-delay-usec = 0

high-priority = no
realtime-scheduling = no
resample-method = trivial
default-sample-rate = 48000
```

Výpis A.4: Skript tlacitko.py

```
import RPi.GPIO as GPIO
import time

BUTTON = 17

GPIO.setmode(GPIO.BCM)
GPIO.setup(BUTTON, GPIO.IN)

while True:
    state = GPIO.input(BUTTON)
    if state:
        print("off")
    else:
        print("on")
time.sleep(1)
```

Výpis A.5: Skript runscript.sh

```
#!/bin/bash

ID="sip:zvonek1@sip2sip.info"
REGISTRAR="sip:sip2sip.info"
REALM="sip2sip.info"
USERNAME="zvonek1@sip2sip.info"
PASSWORD="zvonek123456"
CONTACT="sip:zvonek1@sip2sip.info"
OUTBOUND="sip:proxy.sipthor.net:5060"
THREAD="1"
NAMESERVER="8.8.8.8"
NAMESERVER2="8.8.4.4"
CLOCKRATE="8000"
ADDCODEC="pcma"
DISCODEC="opus/48000/2"
DISCODEC2="AMR-WB/16000/1"
DISCODEC3="AMR/8000/1"

python copyfile.py
python registration.py --id sip:zvonek1@sip2sip.info
--registrar sip:sip2sip.info --realm sip2sip.info
--username zvonek1 --password zvonek123456
--contact sip:zvonek1@sip2sip.info
--outbound sip:proxy.sipthor.net:5060 --thread-cnt 1
--nameserver 8.8.8.8 --nameserver 8.8.4.4 --publish
--clock-rate 8000 --add-codec pcma
--dis-codec opus/48000/2 --dis-codec AMR-WB/16000/1
--dis-codec AMR/8000/1  --client sip:zvonek2@sip2sip.info
--client2 sip:zvonek3@sip2sip.info
```

Výpis A.6: Skript copyfile.py

```
import subprocess as sub
import shlex

process = sub.Popen(shlex.split("sudo␣cp␣/etc
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣/wpa_supplicant/wpa_supplicant.conf
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣/boot"),
                     stdout=sub.PIPE,
                     universal_newlines=True)

while True:
    output = process.stdout.readline()
    print((output.strip()))

    return_code = process.poll()
    if return_code is not None:
        print('RETURN␣CODE', return_code)
        for output in process.stdout.readlines():
            print(output.strip())
        break
```

Výpis A.7: Skript call.py

```
import sys
import pjsua as pj
import argparse
import RPi.GPIO as GPIO

BUTTON = 17
GPIO.setmode(GPIO.BCM)
GPIO.setup(BUTTON,GPIO.IN)

class ParseArgumennts():
    def parse(self,args=sys.argv[1:]):
        parser= argparse.ArgumentParser()


        parser.add_argument("--id")
        parser.add_argument("--registrar")
        parser.add_argument("--realm")
        parser.add_argument("--username")
        parser.add_argument("--password")
        parser.add_argument("--contact")
        parser.add_argument("--outbound")
        parser.add_argument("--thread-cnt")
        parser.add_argument("--nameserver", action="append")
        parser.add_argument("--publish", action="store_true")
        parser.add_argument("--clock-rate")
        parser.add_argument("--add-codec")
        parser.add_argument("--dis-codec", action="append")
        parser.add_argument("--client")
        parser.add_argument("--client2")
        options = parser.parse_args(args)
        return options


LOG_LEVEL=3
current_call = None

# Logging callback
def log_cb(level, str, len):
```

```
    print str,


# Callback to receive events from account
class MyAccountCallback(pj.AccountCallback):

    def __init__(self, account=None):
        pj.AccountCallback.__init__(self, account)

    # Notification on incoming call
    def on_incoming_call(self, call):
        global current_call
        if current_call:
            call.answer(486, "Busy")
            return

        print "Incoming␣call␣from␣", call.info()
        .remote_uri
        print "Press␣'a'␣to␣answer"

        client1=args.client
        client2=args.client2
        if call.info().remote_uri == client1:
            current_call = call
            call_cb = MyCallCallback(current_call)
            current_call.set_callback(call_cb)
            current_call.answer(180)
        else call.info().remote_uri == client2
            current_call = call
            call_cb = MyCallCallback(current_call)
            current_call.set_callback(call_cb)
            current_call.answer(100)


# Callback to receive events from Call
class MyCallCallback(pj.CallCallback):

    def __init__(self, call=None):
        pj.CallCallback.__init__(self, call)
```

```
    # Notification when call state has changed
    def on_state(self):
        global current_call
        print "Call␣with", self.call.info().remote_uri,
        print "is", self.call.info().state_text,
        print "last␣code␣=", self.call.info().last_code,
        print "(" + self.call.info().last_reason + ")"

        if self.call.info().state ==
        pj.CallState.DISCONNECTED:
            current_call = None
            print 'Current␣call␣is', current_call

    # Notification when call's media state has changed.
    def on_media_state(self):
        if self.call.info().media_state ==
        pj.MediaState.ACTIVE:
            # Connect the call to sound device
            call_slot = self.call.info().conf_slot
            pj.Lib.instance().conf_connect(call_slot, 0)
            pj.Lib.instance().conf_connect(0, call_slot)
            print "Media␣is␣now␣active"
        else:
            print "Media␣is␣inactive"

    def on_dtmf_digit(self,digits):
        print(digits)

# Function to make call
def make_call(uri):
    try:
        print "Making␣call␣to", uri
        return acc.make_call(uri, cb=MyCallCallback())
    except pj.Error, e:
        print "Exception:␣" + str(e)
        return None
```

```
parser = ParseArgumennts()
args = parser.parse()

lib = pj.Lib()

try:

    lib.init(log_cfg = pj.LogConfig(level=LOG_LEVEL,
                                    callback=log_cb))


    transport = lib.create_transport(pj.TransportType.UDP,
                                     pj.TransportConfig(0))
    print "\nListening on", transport.info().host,
    print "port", transport.info().port, "\n"


    lib.start()


    acc = lib.create_account(pj.AccountConfig
    (username=args.username, password=args.password,
     domain=args.realm, proxy=args.outbound))


    if len(sys.argv) > 1:
        lck = lib.auto_lock()
        current_call = make_call(sys.argv[1])
        print 'Current call is', current_call
        del lck

    my_sip_uri = "sip:" + transport.info().host + \
                 ":" + str(transport.info().port)

    # Menu loop
    while True:
        print "My SIP URI is", my_sip_uri
        input = GPIO.input(BUTTON)
        if input :
```

```
            lck = lib.auto_lock()
            current_call = make_call(address = args.client)
            del lck


    # Shutdown the library
    transport = None
    acc.delete()
    acc = None
    lib.destroy()
    lib = None

except pj.Error, e:
    print "Exception: " + str(e)
    lib.destroy()
    lib = None
```